JPL

Design SUPERCON97

Case Study of the Design of a Viterbi Decoder

Dr. Gary R. Burke

Jet Propulsion laboratory
California 1 nstitute of Technology
4800"0211< Grove 1 Drive
PasadenaCA 91109-9099

1997
On-Chip System
Design Conference

Abstract

This paper describes the design of an ASIC performing the Viterbi decoder function with a constraint length of 15 and at a speed of 4.4 Mb/s. The architecture of the chip is described, as well as the design challenges which were overcome. Some of the challenges included a large amount of I/O, application of low-noise interconnection scheme to reduce the ground noise of the system, and low - voltage signaling, high chip complexity at 970k transistors for logic and 64k bits of RAM, transfer of data between ASICs at system clock speeds, and internal data path speed at 62 MHz. The author also describes the successful application of the design system

Authors/Speakers

Dr. Gary R. Burke

Jet Propulsion Laboratory
California Institute of
Technology
Pasadena

Current Activities:
ASIC design using
GaAs, CMOS, Gate Arrays,
Compiled Cell

Past Activities:

ASIC and microprocessor design using

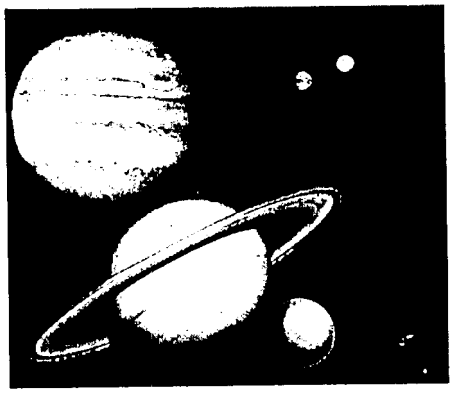
BiCMOS.Std-Cell, 121,
Custom at
Fairchild, Cray, others,

Teaching - IC Design and Computer Science

Gary has a Ph.D from Manchester University, UK

Slide #7

Composite of Solar System



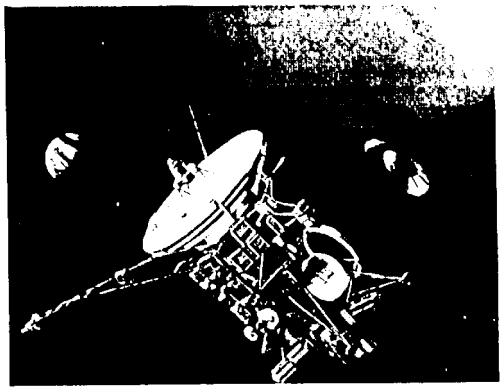
Slide #8

Saturn

Saturn is 9.5AU from the sun, - 883 million miles.

Slide #9

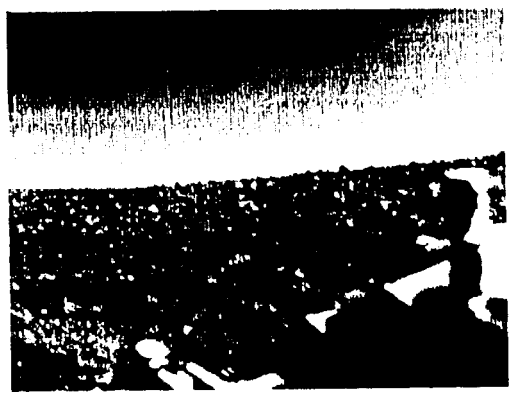
Cassini Spacecraft



The large dish is the high gain antenna, used to transmit data to the Deep Space Network (DSN). Above this is the low gain antenna, which is used to receive commands from the DSN.

Slide #10

Mars



This was taken by the Viking Lander. Mars is 1.5AU from the sun - 139 million miles.

Case Study of the Design of a Viterbi Decoder

Slide #3

Outline

- JPL's role in NASA's space program
- Problems in transmitting data at planetary distances
- The Jetbvd ASIC
- Design approach tools
- Layout and timing issues
- I/O problem
- Clock problem
- Summary/Conclusions

Slide #4

JPL

Vision

Expanding the frontiers of space to enrich knowledge and benefit humanity

Mission

Conduct challenging robotic space missions for NASA

Slide #5

Current Missions

•Voyage	1977
- Outer planets	1972-89
•Galileo	10/1989
- Jupiter probe	12/1995
•Orbit insertion	12/1995
•Ulysses	10/1990
- Sun S. Pole	19942000
- Sun N. Pole	19952001
• Topex/Poseidon	8/1 992
- Earth orbit 18870 orbits to date	

Slide #6

Future Missions

•Pathfinder	12/1996
- Mars landing	7/1997
. Cassini	10/1997
- Saturn Orbit	7/2004
- Probe	11/2004
. Deep Space 1	7/1998
- comet/asteroid Mcaulif	2/1999
- comet/asteroid WK1	6/2000
. Global Surveyor	12/1998
- Mars orbit	9/1999
- Mars Lander	12/1999
•Stardust	2/1 999
- Flyby comet Wild2	1 /2004
- Return to Earth	1/2006

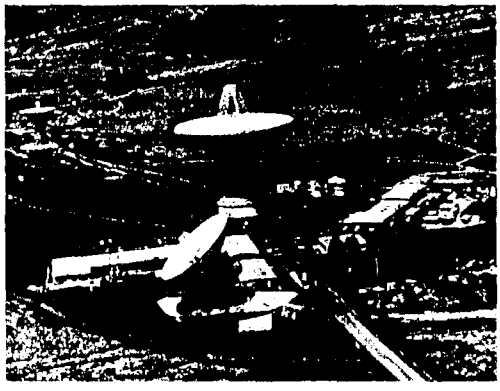
Slide #/1 1

Mars Pathfinder

Pathfinder will land on Mars and release 'rover' which will explore the surface. Telemetry from the Rover is relayed to 1)SN via the lander.

Slide #12

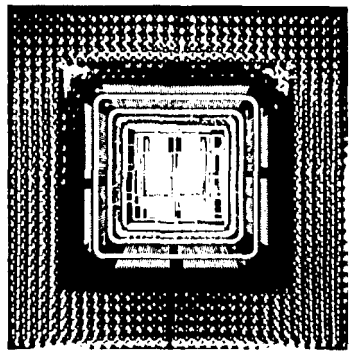
Antennas at DSN Canberra



Signals from the Space- craft arc received at one of three)eep Space4 network (DSN) sites. The photo shows the site in Canberra - Australia. Each site has 6 antenn as, up to 70 meters in diameter. The other sites are in Goldstone (California) and Madrid (Spain), Since it is impractical to increase the size of the antennas, other methods must be used to improve the Bit Error Rate (BER), to ensure signals can be reliably received.

Slide #13

Jetbvd ASIC

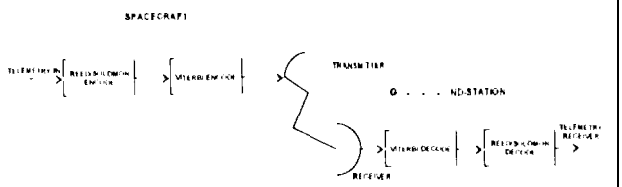


The JETBVD ASIC is designed to decode signals from distant spacecraft.
Characteristics of the ASI C are as follows:

* No of transistors = 970K
* On-chip RAM = 64K
* I)iesize=500X500mils
* Number of 1/() = 508
* Technology = CM OS, 0.55 micron (National Semiconductor)
* Speed = 62MHz
* Package = BGA
* No of ASI (k per system = 64

Slide'#/14

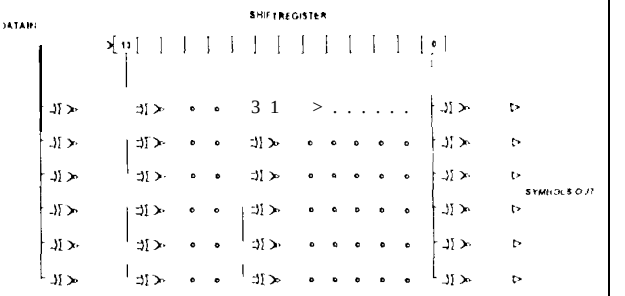
Encoding and Decoding of Spacecraft Signals



The spacecraft signals are encoded to reduce the bit error rate. The outer encoding is Reed/Solomon and the inner code is Viterbi (convolutional encoding).

Slide#/15

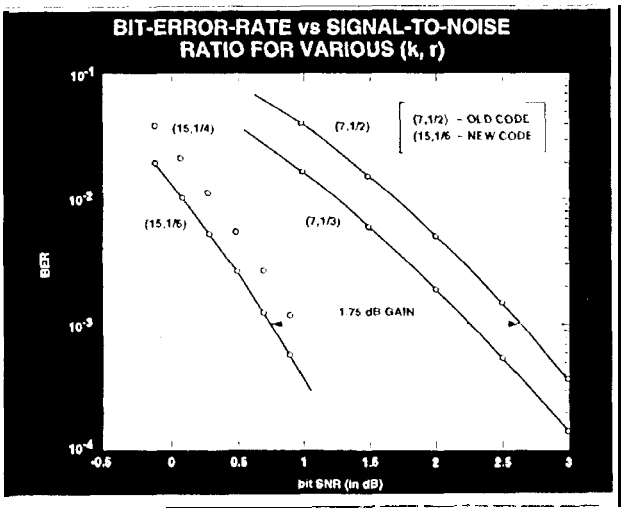
Viterbi Encoder



The concept of Viterbi encoding is to reduce bit error rates by spreading the information over several symbols. For Each Data bit, 1/r symbols are transmitted, where r is the rate (eg 1/6). These symbols are encoded from a shift register with length k-1, where k is the constraint length. Each symbol is formed from exclusive or gates as shown, the number and position of these exclusive or gates is different for each symbol, and is defined in the connection vectors. The exclusive or gates are connected to the shift register. The symbols are actually transmitted sequentially. Each symbol is transmitted as one bit, but when it is received it will have a value which is digitized to 8-bits. This is reduced to 6 bits for the Viterbi Decode operation.

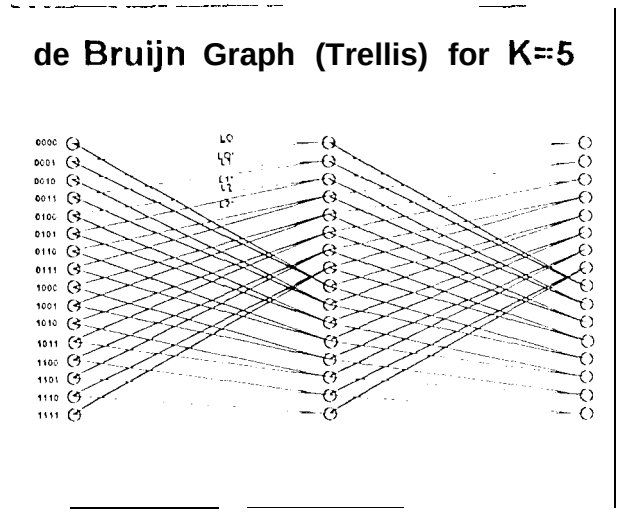
Case' **Study of the**
Design of a Viterbi Decoder

Slide #16



This chart shows how the bit error rate varies for different constraint length k and rate r. A previous standard is the (7 1/2 code). The new code is (15,1/6) which allows a 1.7dB gain.

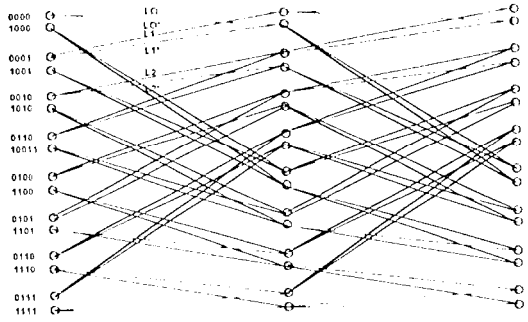
Slide #17



The Viterbi Decoder is considerably more complex than the encoder. The encoding shift register has 2^(k-1) states. For k=5, this is 16 states The probability of being in each state needs to be calculated. This is done in a calculating node. In the picture, the nodes are in a vertical column. The nodes need to communicate to other nodes as shown. The next column are the same set of nodes, repeated for clarity. For example, node 0000 can become 0000 or 1000 depending on the data. The labels L0 L1 etc represent the required symbol pattern in order to transition from one state to the next. The probabilities are calculated as metrics, where a low metric represents a high probability of being in that state.
If the connection vectors are chosen so that they all have a 1 in the lsb, then the labels can be paired as shown, with every the labels from odd nodes being the complement of labels from even nodes.

Slide #/18

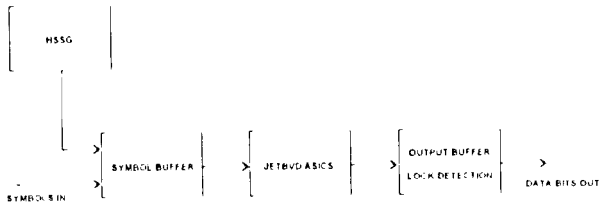
Butterfly Grouping of Nodes



If the nodes are grouped as shown, the labels can be simplified further, with four paths sharing the true and complement of the labels. This is utilized to simplify the logic in the nodes.
For k= 15, the number of nodes is 2^14 or 16K . So that 16K calculations need to be performed in a single bit time. Furthermore, the results of the calculations must be transferred to the successor node in the same bit time. The routing problem is very difficult and results in a large number of interconnects in the system.

Slide #/19

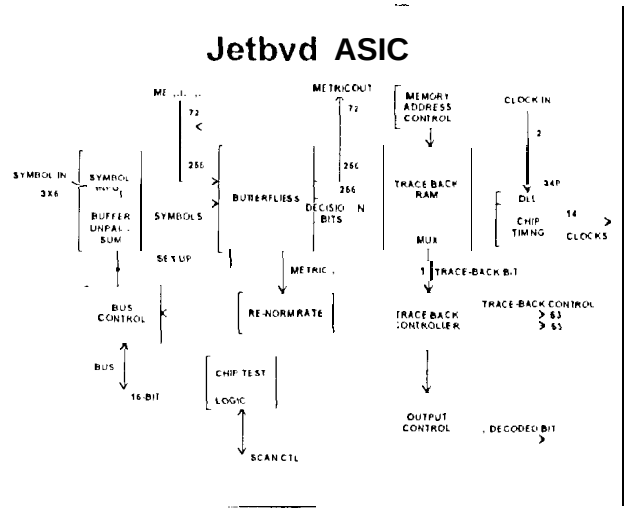
B3MCD Block Diagram



The symbols are buffered and prepared before being sent to the ASIC array for decoding. For test purposes a Symbol Generator can be connected into the input. One of the main function of the support logic is to determine whether the system is in lock. Since there are 6 symbols in a data bit(worse case) there is a 6 fold ambiguity in the position of these symbols relative to the start of a bit. When these are positions correctly, the system is in lock.

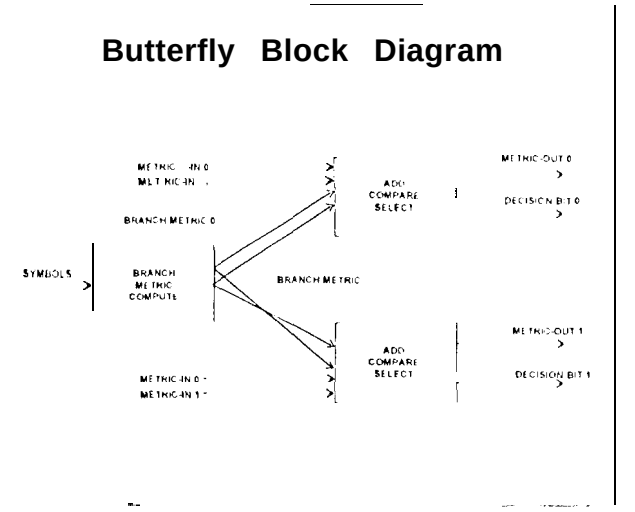
Case Study of the Design of' a Viterbi Decoder

Slide #/20



Signals are input to the Symbol Input Block, where they are buffered and preprocessed. They then go into the butterfly processors, which produce Metrics. The Metrics are fed back to the butterfly processors either on-chip or on other ASICs. The Decision bits from the butterfly processors are then transferred to the RAM. Pointers in the Ram address block keep track of the read and write locations. The Traceback block implements the traceback logic, which reads the! RAM. It outputs traceback bits to connect to the' other ASICs. Data from the traceback block is transferred to the Output Block, which contains LIFOs. The Output Block outputs the data bit in the' correct order.

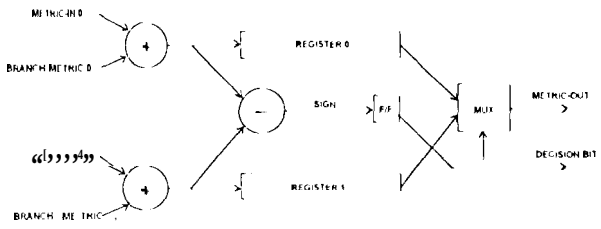
Slide #/21



There are a total of 16K ACS units, in the system. One ASIC contains 256 of these. They are grouped into butterflies, so that two ACS units share a BMC unit. The task of the butterfly is to receive the old values of the accumulated metrics, and based on the new symbols, to produce new values of these metrics, plus decision bits. The decision bits indicate which metric was 'pruned', and corresponds to the data bit, if this node is on the correct path. Later processing (traceback) reads the decision hits to determine the decoded data.

Slide #/22

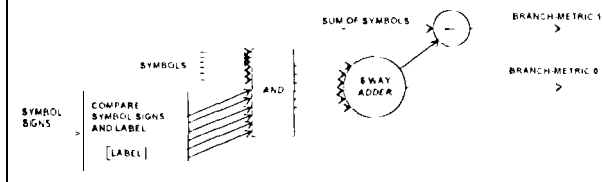
Add Compare Select (ACS)



The Add Compare Select (A(X) unit adds the values of the branch metric (from the BMC) to the incoming metrics, and determines which is the smaller. The smaller of the metrics is allowed to proceed. The decision bit records which of the 2 metrics was chosen. All arithmetic in the ACS is performed serially. The word size is 1 2-bits, but 14 clocks are used to allow for pipeline registers.

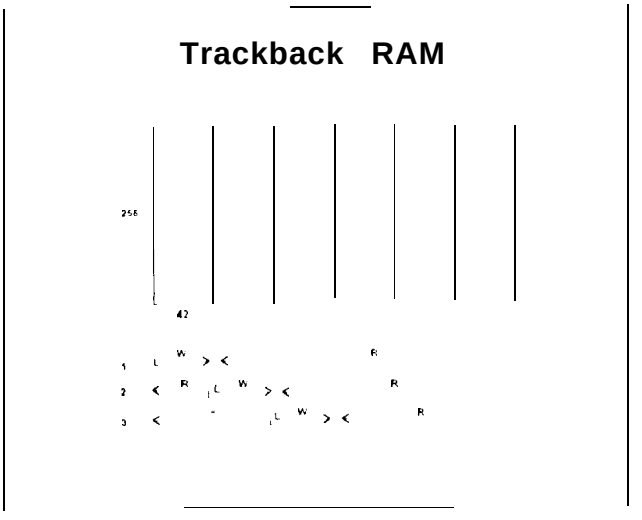
Slide #/23

Branch Metric Compute (BMC)



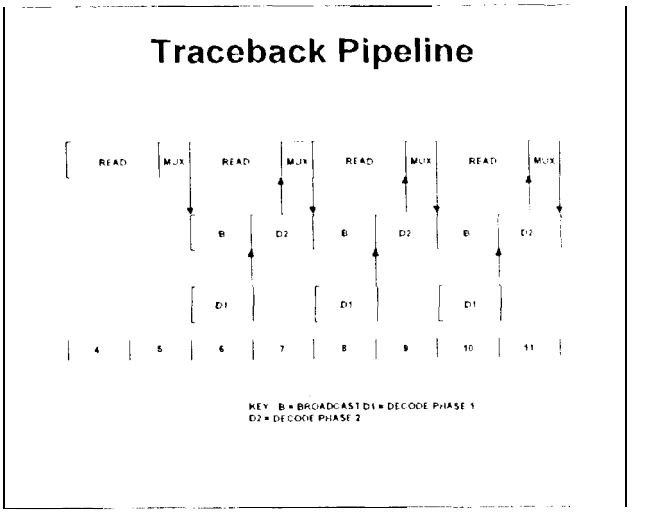
The Branch Metric Compute Block (BMC) calculates the hamming distance between the new symbols and the label stored for that butterfly. It produces a branch metric output, which is greater in value if the match is worse, and 0 in value if the match is perfect. A complementary branch metric is also produced, to simplify the butterfly logic. All arithmetic in the BMC is performed serially.

Slide #24



The other major component of the JETBVD chip is the Traceback Ram. This is where the decision bits are stored. The RAM is 256 words of 252 bits. It can be conceptually thought of as being divided into 6 blocks. While 42 bits are being written into one block, the other 5 blocks are being read.

Slide #25

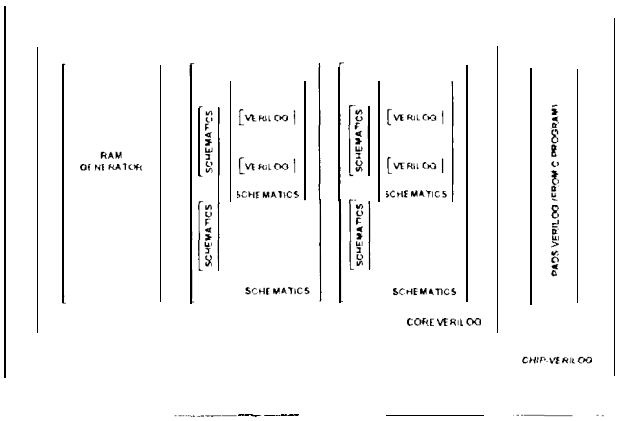


From the traceback RAM the word read (242 bits) is muxed to one bit. The mux address is supplied from a shift register (Node Address - NA), which is fed from the data bit read. So the next bit read out of the ram depends on the previous bit.
Additionally, since the traceback ram is spread over all 64 ASICs, the bit read from the ram has to be broadcast. An additional complication is that the new mux address is decoded from the NA. This decoding takes into account the re-mapping of the nodes onto each ASIC.
Since 5 read accesses need to take place in 14 clocks, the traceback procedure is pipelined as shown, with each read taking only 2 clocks. An additional 3 clocks are used for the write operation.

Case study of the Design of a Viterbi Decoder

Slide #26

Generating Design

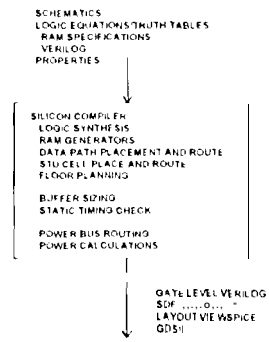


A combination of schematic entry and verilog was used to generate the design. The butterfly processors were grouped to form two wide (64bit) data paths. Elements in this data path (eg registers) were generated using a c program which outputs verilog. This allowed the use of additional elements (eg scan path buffers, clock drivers) in the data path structure.

Some decode logic was generated using Finesse - the logic synthesizer in the Cascade tool. Ram blocks were generated using the ram generator in the silicon compiler. Blocks of schematics and other blocks were connected using verilog. The interconnection pattern for the metrics is very complex and was automatically generated with a c program. It was easier to perform the top level connections using verilog rather than re-enter the connections on a schematic. Similarly, the pads were easier to generate in a program, and output in verilog. The program keeps track of for example numbers of outputs, and inserts power pads as needed.

Slide #27

Compiling the Design



As blocks of the design are complete, they are entered into the silicon compiler. The compiler outputs a verilog netlist with timing information. It also has a built in static timing analyzer. I used this to verify that each block meets the timing spec. Since the block was already place and routed, timing values used are always 'back-annotated' values. They may not be he final values, but are realistic and close to the final values. If the timing analyzer shows some long paths, It is possible to increase the buffer size until the path meets the spec. This can be performed automatically.

Slide #28

Simulation is Performed at 4 Levels

- C model simulation
- Verilog RTL simulation at RTL
- Verilog structural simulation
- Verilog timing simulation

The c model is the behavioral simulation, This model was supplied to us, by another group at JPL. Since it had already been well matched to the theoretical curves, we took this as our 'golden reference'. All results at various levels are compared. Mixed simulation is also performed, with blocks of structural verilog being placed in the RTL simulation, for verification during the design process. The 64-A SIC simulation was also performed using mixed RTL/structural simulation without this simulation would have been impossible. Some changes were necessary to our c-model in order to match the 64-asic simulation.

Slide #29

Generating Vectors

- Scan based design
- Memory self test
- Small number of I/Os test complete design
- ATPG for coverage of logic
- Pads tested through scan path
- Additional functional test vector sets

Approximately 4 million vectors are used for the test, of which all but 250K are scan based vectors. Although we used a full probe card during wafer test, the part was designed to be testable with only the scan controls active. Most I/Os are bidirectional, so that the output/input value can be tested without going to a probe. Most of the fault coverage is gained through the atpg vectors and the memory self test, but some functional vectors are added to improve coverage of 'difficult' areas such as the data bus, and logic on the ram output

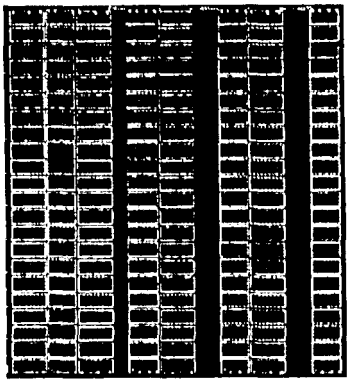
Slide #30

Floor Planning

The placement tool in the floorplanner gives a default placement. I used floorplanner to further refine the placement. The picture shows the 'symbol input' block. This contains a set of pipeline registers which I combined into a data path , some synthesized logic, and various miscellaneous f/f's and gates. The data path is split off automatically, and the rest of the logic is combined into a standard cell block. The aspect ratio of the standard cell block is alterable, and it can be rotated as needed. Ports can be moved individually or in groups to optimize the layout.

Slide #31

Data Path



The data path compiler places the data path horizontally, with each horizontal row representing a bit of the data path. This picture is the data path from the symbol input block. The gaps in the path are to allow routing out of the data path.

Slide #32

Timing Analysis

The timing analysis tool helps to solve both short path and long path problems. Short path problems occur when the clock skew is larger than the path delay, so that an incoming signal clews not have time to be latched. Usually, this causes a hold time violation, which can be caught by the timing analysis u.sine minimum delay parameters. 1 ,ong path delays can be found be displaying the worse delay case paths through the block. This is performed using worse case delay parameters. The Timing Analysis tool is '1'attic - part Of the Cascade tool set.

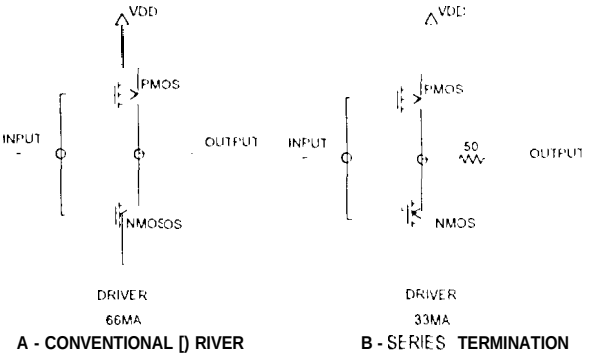
Slide #33

I/O PROBLEM

- Each ASIC has 133 inputs and 133 outputs
- All ASICS are interconnected
- Only 1 clock allowed for I/O transfer
- Conventional Output can draw 66mA peak during switching
- Switching current per ASIC = 8.8Amps
- Switching current per board = 561 Amps!

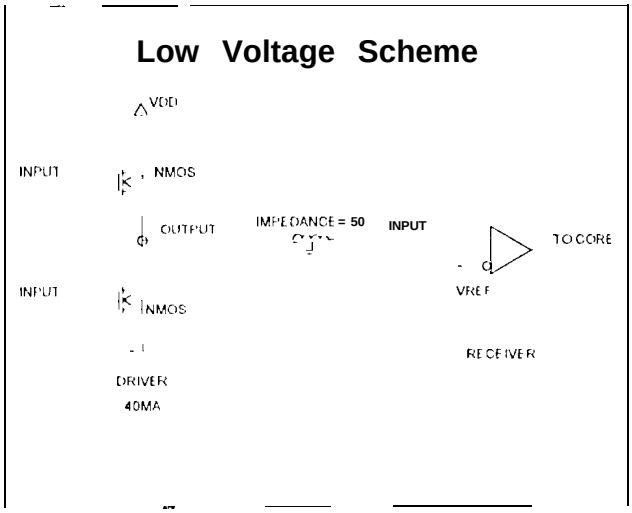
Slide #34

I/O Scheme



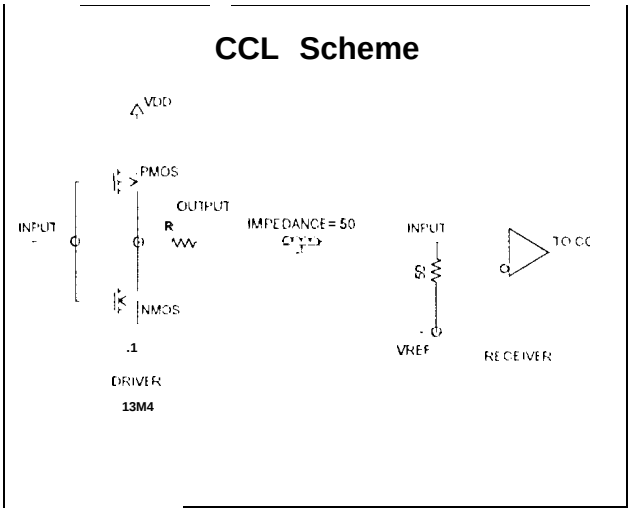
A simplified determination of the peak current, ignoring capacitance effects, is to determine how much current is required to drive the transmission line at the output of the circuit, this is VDD/Z0. In our case VDD= 3.3v and Z0 = 50ohms. The peak current is therefore 66mA.
There are ways to reduce this peak current using slew rate control. However these slow the output. One method to improve this is to serially terminate the line at the source. By terminating with 50 ohms, reflections are reduced and the current is reduced to 33mA.

Slide #35



Reducing the voltage swing will reduce the current. At 2v the current reduces to 40mA. A comparator is needed at the receiving input to detect the signal. Since ground noise is directly coupled to the signals, so that noise margin is impaired.

Slide #36

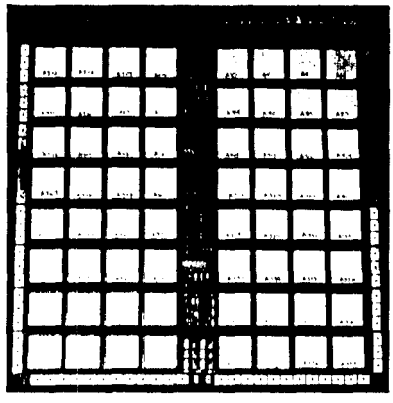


In the new scheme (CCL) the signaling current can be selected independently of the line impedance and the value of VDD. Furthermore CCL is resistant to transients on the power lines. A source resistor (R) limits the current to the desired value . For example, a R value of 115 ohms gives a max current if 13mA and a 1 volt swing at the input resistor. The line is matched at the receiver by a shunt resistor to VDD/2 (1.65 volts). A comparator measures the voltage across the resistor. The combination of the resistor and the comparator is equivalent to a current sensor.

The value of R is not critical, since it does not have to match the line impedance. In the case of the JETBVD ASIC, this resistor was in effect combined with the output transistors, by adjusting the size of these transistors give the required current.

Slide #37

Decoder Board



Sixty four highly connected AS I Cs are on the decoder board.

Slide #38

Clock Skew Calculation

C1 = Destination clock time

C2 = Source clock time

T = f/f to f/f delay ASIC to ASIC

Short Path Problem

C1(max) - C2(min) > T(min)

Long Path problem

C1'(min) - C2(max)< T(max)

Clock skew between devices can arise from process and temperature variations and voltage variations across the board.

Slide #39

7.3 Minimize Clock Skew

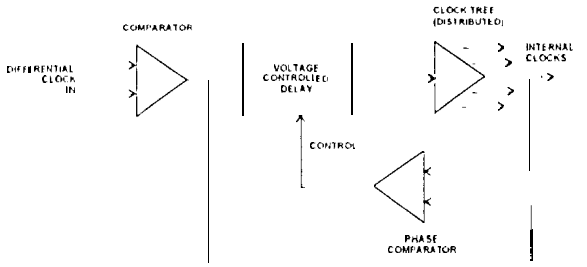
- Individually adjust external clocks
  - Complex
- 1 low skew high power clock driver per ASIC
  - not feasible
- On chip PLL
- On chip DLL
  - Simpler than PLL

With such a large number (64) ASI Cs on the system board, it is important to keep clock skew to a minimum. This is accomplished on the JETBVD ASI C by the use of a 1)elay Lock Loop (1)1,1,) on every ASI C. The DLL adjusts the internal clock until it is in-phase with the external clock. Inside the ASI C a clock tree fanout from the 1)LL is arranged to balance clock delays across the ASI C. Since one branch of this tree is the feedback to the DLL, skew due to clock tree delay variation from ASIC to ASIC is very small (less than 1V),

Case Study of the Design of a Viterbi Decoder

Slide #40

Delay Locked Loop



Slide #41

Jetbvd Layout

The ASIC is packaged in a 520 ball BGA. The ASIC size is approx 500 X 500 roils. The photo shows the finished layout with the main data paths and RAM clearly visible.

Slide #42

Design tools

Cascade	Epoch Silicon Compiler
Cadence	Concept Schematic Entry Verilog Simulation
Sunrise	ATPG
Zycad	Verilog simulation accelerator Fault grading
Metasoftware	Hspice - circuit simulation
C/Pascal	Behavioral modeling etc.

Slide #43

Results

- DLL works
- 1/0 scheme works
- Viterbi Decoder Works
- First Pass Success

Slide #44

Summary

Problem	Solution
Huge SS0 problem	CCL I/O
Clock skew	DLL
Layout/timing	Silicon Compiler
Test vectors	Scan logic/ atpg

Slide #45

Acknowledgments

The author wishes to acknowledge tho assistance of Bill Whitaker, who performed most of the simulations, and the valuable help of the MCD3 dosign team: Jeff Buchanan, Scott Graham, Becky Heninger, Robert Johnson, Jim Kowalski, Sho Nakahira, Ramona Tung, Joe Weisberger.

Tho work described was performed at the Jot Propulsion Laboratory, California Institute of Technology under contract with the National Aeronautics and Space Administration.